ELSEVIER Journal of Computational and Applied Mathematics 59 (1995) 349–380

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

Incomplete partial fractions for parallel evaluation of rational matrix functions

D. Calvetti[a,1], E. Gallopoulos[b,2], L. Reichel[c,*,3]

[a] *Department of Pure and Applied Mathematics, Stevens Institute of Technology, Hoboken, NJ 07030, United States*

[b] *Coordinated Science Laboratory, Center for Supercomputing Research and Development, University of Illinois at Urbana-Champaign, Urbana, IL 61801, United States*

[c] *Department of Mathematics and Computer Science, Kent State University, Kent, P.O. Box 5190, OH 44242-0001, United States*

Dedicated to Richard S. Varga on the occasion of his 65th birthday

Received 13 January 1994; revised 22 March 1994

Abstract

Frequently, one needs to evaluate expressions of the form $[p(A)]^{-1}q(A)b$, where $A \in \mathbb{R}^{N\times N}$, $b \in \mathbb{R}^N$, and p and q are polynomials with degree $q \leqslant$ degree p, and such that no zero of p is an eigenvalue of A. Algorithms based on the partial fraction representation of q/p when evaluating $[p(A)]^{-1}q(A)b$ lend themselves well to implementation on a parallel computer, but might yield poor accuracy. We discuss how to determine an incomplete partial fraction representation of q/p which allows parallel computation, while retaining high accuracy.

Keywords: Partial fraction representation; Parallel algorithm; Cyclic reduction; Partial differential equation; Roundoff error analysis

1. Introduction

The present paper is concerned with the problem of accurate evaluation of

$$x := [p(A)]^{-1}q(A)b, \tag{1}$$

when $A \in \mathbb{R}^{N\times N}$, $b \in \mathbb{R}^N$, p and q are polynomials. Throughout this paper we assume that

* Corresponding author. E-mail: reichel@mcs.kent.edu.

[1] Research supported in part by the Design and Manufacturing Institute of Stevens Institute of Technology.

[2] Research supported in part by the National Science Foundation under NSF grant CCR-9120105.

[3] Research supported in part by NSF grant DMS-9205531.

SSDI 0377-0427(94)00037-2

- the degree m of q is not larger than the degree n of p,
- the polynomial $p(t) = a_0 \prod_{j=1}^{n} (t - t_j)$ has distinct zeros t_j, none of which is an eigenvalue of A, and
- the polynomials p and q are relatively prime.

Assume for the moment that $q := 1$. Then the traditional way of evaluating (1) is given by the following algorithm:

Algorithm 1.1. Compute solution x of (1) when $q := 1$.

1. $x_0 := b$;
2. for $j := 1, 2, \ldots, n$ do
   solve $(A - t_j I)x_j = x_{j-1}$;
   endfor
3. $x := a_0^{-1} x_n$;

Algorithm 1.1 uses explicitly the product form representation $p(t) = a_0 \prod_{j=1}^{n} (t - t_j)$. This representation has several advantages compared with the common power form representation $p(t) = \sum_{j=0}^{n} c_j t^j$, such as the matrix $p(A)$ does not have to be explicitly computed, and the product form representation of a polynomial is generally better conditioned than the power form; for example, see [40].

An important observation is that in Algorithm 1.1, n linear systems of equations have to be solved sequentially. Hence, when mapping Algorithm 1.1 on a parallel architecture, one can only exploit the parallelism available in the solution of each individual linear system.

An alternative approach to the evaluation of (1) is based on an application of the partial fraction representation of the rational function. We write

$$\frac{q(t)}{p(t)} = \alpha_0 + \sum_{j=1}^{n} \alpha_j (t - t_j)^{-1}, \qquad \alpha_0 = \lim_{t \to \infty} \frac{q(t)}{p(t)}, \qquad \alpha_j = \frac{q(t_j)}{p'(t_j)}. \tag{2}$$

This representation leads to the following algorithm for computing x:

Algorithm 1.2. Compute solution x of (1) by using formula (2).

1. Compute $\alpha_0 := \lim_{t \to \infty} q(t)/p(t)$, $\alpha_j := q(t_j)/p'(t_j)$, $1 \leqslant j \leqslant n$.
2. Solve $(A - t_j I)x_j = b$, $1 \leqslant j \leqslant n$.
3. Compute $x := \alpha_0 b + \sum_{j=1}^{n} \alpha_j x_j$.

Algorithm 1.2 lends itself better to mapping on a parallel architecture than Algorithm 1.1 because it offers two levels of parallelism: large grain parallelism because n linear systems of equations can be solved in parallel, and medium grain parallelism from the solution of each of the linear systems. The presence of two levels of parallelism is of particular importance when the computer architecture offers some form of hierarchical parallelism. Examples of such architectures include clusters of multiprocessor workstations, massively parallel machines with vector processing units and actual machine prototypes [27]. By mapping each large grain task on a cluster of several tightly coupled processors, communication requests within each task can be serviced efficiently, while there is less demand for costly communication between processor clusters.

When comparing algorithms, the issue of numerical behavior, e.g., error propagation, must take precedence over computational efficiency. Several computed examples in Sections 3 and 5 illustrate that Algorithm 1.2 can be sensitive to roundoff errors. Algorithm 1.1 is less sensitive to perturbation than Algorithm 1.2. This depends on the fact that Algorithm 1.1 uses the product from representation of the denominator p, whereas Algorithm 1.2 is based on the partial fraction representation.

We can easily identify the numerical difficulties of Algorithm 1.2 by noting that the partial fraction coefficients for (2) are

$$\alpha_j = \frac{q(t_j)}{a_0 \prod_{k=1, k\neq j}^{n} (t_j - t_k)}, \quad 1 \leqslant j \leqslant n. \tag{3}$$

It follows that the presence of close poles t_k may cause some of the coefficients α_j to be of very large magnitude. This generally causes numerical difficulties, since a coefficient α_j of large magnitude amplifies the error present in the vector x_j when forming $\alpha_j x_j$ in Step 3 of Algorithm 1.2. Furthermore, if the norm of x is small compared to the norm of the vectors $\alpha_j x_j$ or $\alpha_0 b$, then cancellation of significant digits takes place in Step 3 of Algorithm 1.2, and this can lead to complete loss of accuracy in the computed solution x.

In this paper we explore the numerical problems associated with the application of partial fractions for evaluating (1), and we show how some of the difficulties associated with partial fractions can be remedied by using an *incomplete partial fraction* (IPF) representation of q/p as a basis for algorithms for evaluating (1). The IPF representation amounts to writing

$$\frac{q(t)}{p(t)} = \prod_{l=1}^{\mu} \frac{q_l(t)}{p_l(t)}, \tag{4}$$

and using a partial fraction representation for each of the factors q_l/p_l. Here p_l and q_l are polynomials such that the degree of q_l is not larger than the degree of p_l for each l. We would like to choose the polynomials p_l and q_l so that:

- the partial fraction coefficients $\{\alpha_{jl}\}_{j=1}^{k_l}$ of each factor q_l/p_l, for $1 \leqslant l \leqslant \mu$, are not so large as to cause numerical problems, and
- the number of factors, μ, is small in order to enable efficient parallel evaluation.

This paper is organized as follows. Section 2 provides a brief survey of applications of partial fraction representations in numerical methods. A few simple examples in which the evaluation of the partial fraction representation of q/p yields poor accuracy are described in Section 3, and these examples suggest an approach for determining IPF representations with controlled size of the partial fraction coefficients. We present three algorithms for computing incomplete partial fraction representations based on this approach; the simplest algorithm is for the case when q is a constant. The other algorithms are for the cases when p and q are of the same degree, and when q is of strictly smaller degree than p. Section 4 presents an analysis of one of these algorithms, and computed examples in Section 5 illustrate the performance of the algorithms. In particular, in Section 5.2 we show how IPF representations can be applied to the computation of high-order approximations of the product of a matrix exponential and a vector. Section 6 contains concluding remarks.

2. Previous work

The potential for parallel implementation of partial fraction representations was first pointed out in [28]. Ten years later Sweet [50, 51] proposed the use of the partial fraction representation of reciprocal polynomials in the implementation of the block cyclic reduction (BCR) method on parallel computers. On sequential computers the BCR method is implemented by using the factored form of reciprocal polynomials (e.g., see [6, 36, 47]), but this representation can be difficult to implement efficiently on multiprocessors. The use of partial fractions makes the BCR method competitive with parallel elliptic solvers based on matrix decomposition. The partial fraction approach was also used in [16, 17, 19, 48], where extensions and generalizations of Sweet's work, together with experiments and implementations on multiprocessor architectures are presented and discussed. Recently, the partial fraction representation has been applied to the design of preconditioners [45]. Based on work by Varga and collaborators [8–10, 53, 55, 56], partial fraction representations of rational approximants of the exponential function have been used in the solution of time-dependent problems in [26, 29, 39, 46, 49, 60–63], as well as implicitly in [25]. Partial fraction representations of very high-order rational Chebyshev and Padé approximants of the matrix exponential are advocated in [18, 20, 21, 30]. Applications of the partial fraction representation in algorithms for problems in Control Theory are described in [5, 11, 35]. A parallel version of the rational Krylov algorithm for generalized eigenvalue problems described in [41] also uses the partial fraction representation of rational functions.

Other algorithms based on the partial fraction representation are considered in [60, 61, 63].

The possible loss of accuracy in the evaluation of (1) using partial fractions has been noted before; see [20, 35, 46, 59, 64]. Swarztrauber [47] observes that in the BCR method a reduction in the arithmetic work required for the evaluation of expressions of the form (1) when both p and q are of degree n can be achieved by replacing the computation of x_{j+1} by

$$(A - t_j I)x_{j+1} = (A - s_j I)x_j, \tag{5}$$

with the determination of x_{j+1} according to

$$x_{j+1} = x_j + (t_j - s_j)(A - t_j I)^{-1}x_j, \tag{6}$$

for $j = 1, 2, \ldots, n$, where the s_j denote the zeros of the numerator polynomial q in (1), and as usual the t_j denote the zeros of the denominator polynomial p. Although not explicitly stated in [47], formula (5) uses the partial fraction representation of the rational function $(t - s_j)/(t - t_j)$. In order to reduce the influence of propagated roundoff errors, Swarztrauber [47] suggests that the s_j and t_j be ordered so as to keep the differences $|t_j - s_j|$ small for all j. This amounts to selecting pairs $\{s_j, t_j\}$ so that the coefficients of the partial fraction representation of $(t - s_j)/(t - t_j)$ are small. An analysis of the behavior of roundoff errors when using Algorithm 1.1 in the BCR method can be found in [36].

Algorithms for computing incomplete partial fraction representations have previously been considered by Henrici [23]. These algorithms are designed to compute the coefficients of an IPF representation efficiently once the partial factors have been decided upon, rather than determining partial factors that yield IPF representations with small coefficients.

3. Incomplete partial fraction representations

Let r denote the product form representation (4) of the rational function q/p. Given r and a threshold $\tau > 0$, we want to design algorithms for the generation of rational functions $r_l = q_l/p_l$, $1 \leqslant l \leqslant \mu$, that satisfy:

- $r(t) = \prod_{l=1}^{\mu} r_l(t)$,
- the degree of q_l is not larger than the degree k_l of p_l for each l,
- the polynomials p_l and q_l are relatively prime for each l, and
- the coefficients α_{jl} of the partial fraction representation of r_l satisfy $|\alpha_{jl}| \leqslant \tau$ for each l.

We refer to an incomplete partial fraction representation that satisfies these requirements as IPF(τ). For any given τ, there may be more than one representation satisfying the above requirements. Among them it is preferable to use representations with a small number of factors μ.

Let $\{t_{jl}\}_{j=1}^{k_l}$ denote the zeros of p_l. We then write the IPF (τ) representation as

$$\frac{q(t)}{p(t)} = \prod_{l=1}^{\mu} \left(\alpha_{0l} + \sum_{j=1}^{k_l} \frac{\alpha_{jl}}{t - t_{jl}} \right), \qquad \sum_{l=1}^{\mu} k_l = n. \tag{7}$$

Once the partial fraction representation of each factor r_l has been determined, the evaluation of the right-hand side of (1) can be carried out as follows:

Algorithm 3.1. Compute solution x of (1) using IPF(τ) representation (7).

Input: $b \in \mathbb{R}^N$, $A \in \mathbb{R}^{N \times N}$, $\bigcup_{l=1}^{\mu} \{t_{jl}\}_{j=1}^{k_l}$, $\bigcup_{l=1}^{\mu} \{\alpha_{jl}\}_{j=0}^{k_l}$, where $\sum_{l=1}^{\mu} k_l = n$.

Output: $x := p(A)^{-1} q(A) b$.

1. $x := b$;
2. for $l = 1, \ldots, \mu$ do
   2.1. Solve $(A - t_{jl} I) x_j = x$, $1 \leqslant j \leqslant k_l$.
   2.2 Compute $x := \alpha_{0l} x + \sum_{j=1}^{k_l} \alpha_{jl} x_j$.
   endfor

Algorithm 3.1 contains Algorithms 1.1 and 1.2 as special cases. The threshold parameter τ plays a crucial role in the computation. Let q/p be a rational function that satisfies the assumptions stated in Section 1 after formula (1). There is a value $\tau_{\max}$, such that for all $\tau \geqslant \tau_{\max}$, the IPF($\tau$) representation of q/p is the partial fraction representation. In this case Algorithm 3.1 is equivalent to Algorithm 1.2. On the other hand, when $\tau = 0$, then the IPF(τ) representation of q/p is the product form representation. Algorithm 3.1 evaluates this representation using formula (6).

Roundoff errors in the evaluation of (1) by means of an IPF representation (7) can arise in the computation of each of the quantities $x_j = (A - t_{jl} I)^{-1} x$, in the computation of the coefficients α_{jl} of the IPF representation, and when forming the linear combinations $\alpha_{0l} + \sum_{j=0}^{k_l} \alpha_{jl} x_j$. Although problems arising from the propagation of roundoff errors can be significant in the scalar case ($N = 1$), they are greatly accentuated in the context of matrix arithmetic ($N > 1$). In particular, the vectors x_j, which are solutions of linear systems of equations, can be largely affected by roundoff errors. Moreover, if x_j is computed by an iterative method, then the error in x_j may also depend on the stopping criterion chosen for the iterative method. The presence of large partial fraction coefficients can amplify the error in the computed vectors x_j, causing further loss of accuracy.

Therefore, we want to choose the factors r_l so that their partial fraction coefficients α_{jl} are bounded in order to control error propagation.

Many iterative methods for the solution of linear systems of equations determine an approximate solution in a Krylov subspace determined by the matrix. We note that the Krylov subspaces determined by the matrices $A - t_{jl}I$ are independent of the constants t_{jl}. This observation has spurred the development of several iterative methods for the solution of the linear systems of equations in Step 2.1 of Algorithm 3.1; see [5, 11, 14].

3.1. Examples

We now present several examples that illustrate that the evaluation of appropriately chosen IPF representations of rational functions q/p can give much higher accuracy than the evaluation of partial fraction representations. We observe that, in general, the presence of very close poles of the rational function can greatly compromise the accuracy of the computed value of x. A somewhat surprising behavior is illustrated in Examples 3.4 and 3.5, namely that there are cases where the presence of nearby poles does not cause severe loss of accuracy. A study of the conditions under which such behavior occurs is suggestive for the design of algorithms for determining suitable incomplete partial fraction representations.

Example 3.2. Let $p(t) := (t - \frac{1}{2})(t - \varepsilon)(t + \varepsilon)$ with $|\varepsilon| > 0$ tiny. The evaluation of the partial fraction representation

$$\frac{1}{p(t)} = \frac{(\frac{1}{2} - \varepsilon)^{-1}(\frac{1}{2} + \varepsilon)^{-1}}{t - \frac{1}{2}} - \frac{\varepsilon^{-1}(1 - 2\varepsilon)^{-1}}{t - \varepsilon} + \frac{\varepsilon^{-1}(1 + 2\varepsilon)^{-1}}{t + \varepsilon} \tag{8}$$

in finite precision arithmetic can give poorer accuracy than when evaluating the incomplete partial fraction representation

$$\frac{1}{p(t)} = \left(\frac{(\frac{1}{2} + \varepsilon)^{-1}}{t - \frac{1}{2}} - \frac{(\frac{1}{2} + \varepsilon)^{-1}}{t + \varepsilon} \right)(t - \varepsilon)^{-1}. \tag{9}$$

For instance, assume that the computations are carried out with three significant digits and that $\varepsilon := 1/900$. Evaluation of the right-hand side of formula (8) at $t = 1$ then yields the value 8.00, while evaluation of the right-hand side of (9) yields the value 1.99. In exact arithmetic we have

$$\frac{1}{p(1)} = \frac{2}{1 - \varepsilon^2} = 2.0000.$$

This example illustrates that the use of an IPF representation can yield much higher accuracy than a partial fraction representation. The reciprocal polynomial $1/p(t)$ has close poles, but in the IPF representation (9) the partial fraction representation of a reciprocal polynomial with distant poles is determined. This explains why formula (9) yields higher accuracy than formula (8).

Example 3.3. This example shows that the magnitude of the coefficients in the partial fraction representation depends not only on the distance between poles, but also on their distribution. Let

$p(t) := \prod_{j=1}^{n}(t - j/n)$. Then

$$\frac{1}{p(t)} = \sum_{j=1}^{n} \frac{\alpha_j}{t - j/n}, \qquad \alpha_l = \frac{n^{n-1}}{\prod_{j=1, j\neq l}^{n}(l-j)}. \tag{10}$$

In particular, for $n = 20$, we obtain

$$\alpha_{20} = -\alpha_1 = \frac{20^{19}}{19!} \approx 3 \cdot 10^8.$$

Thus, evaluation of the partial fraction representation (10) can yield severe cancellation of significant digits despite the fact that $1/p$ does not have close poles.

The next two examples show partial fraction representations that have been used successfully in parallel matrix algorithms. These examples provide motivation for the design of our algorithms for determining IPF(τ) representations.

Example 3.4. Consider the solution of a Dirichlet problem for Poisson's equation on a two-dimensional rectangular domain, and discretize the Laplace operator by the standard 5-point stencil. This yields a linear system of equations

$$Mu = f, \tag{11}$$

with a block-tridiagonal matrix M. All diagonal blocks of M are a symmetric tridiagonal matrix, that we denote by A, and the off-diagonal blocks of M are $-I$. Assume that M has $2^k - 1$ diagonal blocks A, for some integer $k \geqslant 1$. Then the linear system (11) can be solved rapidly by the BCR algorithm, as described in [6, 36]. This algorithm requires the solution of linear systems of equations of the form

$$p_n(A)x = b, \tag{12}$$

where $p_n(t) := 2C_n(\frac{1}{2}t)$ and $C_n(t) := \cos(n \arccos(t))$. Thus, up to a scaling factor, $p_n(t)$ is a Chebyshev polynomial of the first kind for the interval $[-2, 2]$. The computations of the BCR algorithm require the solution of systems (12) for polynomials of different degrees, with the highest degree being equal to 2^{k-1}. Let $t_j^{(n)}$ denote the zeros of p_n and let $\alpha_j^{(n)}$ be the partial fraction coefficients of $1/p_n$. Then

$$p_n'(t_j^{(n)}) = C_n'(\tfrac{1}{2}t_j^{(n)}) = (-1)^{j-1} n \Big/ \sin\left(\frac{2j-1}{2n}\pi\right)$$

yields

$$|\alpha_j^{(n)}| = \frac{1}{|p_n'(t_j^{(n)})|} \leqslant \frac{1}{n}, \quad 1 \leqslant j \leqslant n. \tag{13}$$

The distance between some adjacent zeros $t_j^{(n)}$ is fairly small for large n. For instance,

$$|t_1^{(n)} - t_2^{(n)}| = 2\left|\cos\frac{\pi}{2n} - \cos\frac{3\pi}{2n}\right| = 4\left|\sin\frac{\pi}{n}\sin\frac{\pi}{2n}\right| < \frac{2\pi^2}{n^2}.$$

Nevertheless, bound (13) shows that $\max_{1 \leqslant j \leqslant n} |\alpha_j^{(n)}|$ does not grow with n. This suggests that it may be possible to evaluate the partial fraction representation (2) without severe loss of significant digits. Computed examples in [17] show that this, indeed, is the case.

Example 3.5. This example comes from the solution of elliptic problems using the parallel solver described in [19]. Bank and Rose [3, Theorem 2.1] show that the (i,j)th block of the inverse of the block-tridiagonal matrix M introduced in Example 3.4 is of the form $S_v^{-1}(A)S_{i-i}(A)S_{v-j}(A)$ for $j \geqslant i$, and $S_v^{-1}(A)S_{j-1}(A)S_{v-i}(A)$ for $j < i$, where v denotes the number of diagonal blocks, and S_k is a Chebyshev polynomial of the second kind of degree k for the interval $[-2, 2]$, i.e.,

$$S_k(t) := \frac{\sin((k+1)\theta)}{\sin\theta}, \qquad \cos\theta = \tfrac{1}{2}t, \quad -2 \leqslant t \leqslant 2.$$

By symmetry, it suffices to consider the case $j \geqslant i$. The parallel algorithm described in [19] uses the partial fraction representation of $S_v^{-1}(t)S_{j-1}(t)S_{v-i}(t)$. The partial fraction coefficients for this function are

$$\alpha_k^{(ij)} := \frac{\sin(j\theta_k)\sin((v+1-i)\theta_k)}{\sin^2\theta_k S_v'(t_k)}, \quad 1 \leqslant k \leqslant v,$$

where $\theta_k := k\pi/(v+1)$ and $t_k := 2\cos\theta_k$. It is easy to show that

$$|\alpha_k^{(ij)}| = \frac{2}{v+1}|\sin(j\theta_k)\sin((v+1-i)\theta_k)| \leqslant \frac{2}{v+1}, \quad 1 \leqslant k \leqslant v.$$

The distance between the closest poles is

$$|t_1 - t_2| = 2\left|\cos\frac{\pi}{v+1} - \cos\frac{2\pi}{v+1}\right| \leqslant \frac{3\pi^2}{(v+1)^2}.$$

Hence, the partial fraction coefficients remain bounded as v increases, even though the distance between some poles decreases.

Example 3.2 shows that the partial fraction representation of a reciprocal polynomial with nearby poles can have coefficients of large magnitude, and that the magnitude of the coefficients can be reduced considerably by using an incomplete partial fraction representation. We also observe that in Examples 3.4 and 3.5, the coefficients of partial fraction representations grow slowly, or not at all, as the degree of the denominator increases, when the poles are zeros of Chebyshev polynomials of the first or second kind in the interval $[-2, 2]$.

The observation that the partial fraction coefficients of a rational function whose poles are distributed like zeros of a Chebyshev polynomial, are fairly small suggests the following approach to determining an incomplete partial fraction representation of an arbitrary reciprocal polynomial $1/p$. Factor $p = \prod_{j=1}^{\mu} p_l$, so that the zeros of each factor are distributed roughly like zeros of a Chebyshev polynomial, and then determine the partial fraction representation of each reciprocal polynomial $1/p_l$. Such a factorization of p can be determined by ordering the zeros of p like Leja points, introduced below. This ordering method has previously been applied to stabilize interpolation polynomials; see [37].

Let T be a compact set in the complex plane $\mathbb{C}$ such that its complement in $\mathbb{C}\cup\{\infty\}$ is connected and regular for the Dirichlet problem. Edrei [12] and Leja [31] studied sequences of points $t_1, t_2, t_3, \ldots$ with the following property. Let t_1 satisfy

$$|t_1| = \inf_{t\in T} |t|, \quad t_1 \in T, \tag{14}$$

and let the points t_k, $k > 1$, be such that

$$\prod_{j=1}^{k-1} |t_k - t_j| = \sup_{t\in T} \prod_{j=1}^{k-1} |t - t_j|, \quad t_k \in T,\ k = 1, 2, 3, \ldots. \tag{15}$$

The points t_k are generally not determined uniquely by (14) and (15). We call any sequence of points $t_1, t_2, \ldots$ that satisfies (14) and (15) a sequence of Leja points for T, or sometimes just briefly Leja points for T.

Example 3.6. Let $T := \{t: |t| \leqslant 1\}$. A sequence of Leja points for T can be constructed as follows. Let $t_1 := 1$. The next three Leja points are given by $t_2 := -1$, $t_3 := \imath$ or $-\imath$ and $t_4 := -t_3$, where $\imath := \sqrt{-1}$. This illustrates nonuniqueness.

Let the nonnegative integer k have binary representation

$$k = \sum_{j=0}^{\infty} k_j 2^j, \quad k_j \in \{0, 1\}, \tag{16}$$

and define

$$t_{k+1} := \exp\left(\imath\pi \sum_{j=0}^{\infty} k_j 2^{-j}\right). \tag{17}$$

It can be shown that the sequence of points $t_1, t_2, t_3, \ldots$ defined by (16) and (17) is a sequence of Leja points for T. Note that the points t_{k+1} are obtained by *bit*-reversal of the binary representation of k. For every integer $k \geqslant 0$, the points $t_1, t_2, \ldots, t_{2^k}$ are equidistant. Moreover, the points $t_1, t_2, \ldots, t_k$ are uniformly distributed on the unit circle with respect to the density function $\sigma(t) = 1/2\pi$ as $k \to \infty$.

Example 3.7. Let T be the interval $[-2, 2]$. The Leja points for T are uniformly distributed on T with respect to the density function

$$\sigma(t) := \frac{1}{2\pi}(4 - t^2)^{-1/2}, \quad -2 < t < 2.$$

This follows from results of Leja [31, Lemme 1] and Walsh [57]; see [38] for details. Note that the zeros of the Chebyshev polynomials $C_n(\frac{1}{2}t)$ for the interval T are also uniformly distributed with respect to $\sigma(t)$ as n increases. This suggests, in view of Examples 3.4 and 3.5, that coefficients of the partial fraction representation of a reciprocal polynomial $1/p(t) = \prod_{j=1}^{n}(t - t_j)^{-1}$, whose poles t_j are Leja points for $[-2, 2]$, are not very large. Theoretical and numerical results in support of this hypothesis are presented in Sections 4 and 5.

3.2. *Algorithms for determining* IPF(τ) *representations*

We present algorithms for computing IPF(τ) representations of rational functions q/p that satisfy the conditions stated in Section 1 after formula (1). We distinguish three cases depending on the degrees m and n of the numerator and denominator polynomials q and p, respectively. These are: (i) $m = 0$, $n > 0$, (ii) $m = n > 0$, and (iii) $0 < m < n$.

3.2.1. *The case* $m = 0$, $n > 0$

The rational function is of the form $1/p$. Let $T = \{t_j\}_{j=1}^n$ be the set of poles of $1/p$. We order the poles so that they satisfy (14) and (15), and we refer to the ordering so obtained as the Leja ordering. Thus, let $t_{11} \in T$ satisfy $|t_{11}| = \min_{t\in T}|t|$, and let $t_{k1} \in T$ be such that

$$\prod_{j=1}^{k-1} |t_{k1} - t_{j1}| = \sup_{t\in T} \prod_{j=1}^{k-1} |t - t_{j1}|, \quad t_{k1} \in T, \ k = 1, 2, 3, \ldots.$$

Compute the coefficients of the partial fraction representation of $\prod_{j=1}^{k}(t - t_{j1})^{-1}$ from the coefficients of the partial fraction representation for $\prod_{j=1}^{k-1}(t - t_{j1})^{-1}$ for increasing values of k. This process continues as long as the magnitude of the coefficients in the partial fraction representation are bounded by a given threshold τ and $k \leqslant n$. Let the integer $k_1 < n$ be such that all partial fraction coefficients of $\prod_{j=1}^{k_1}(t - t_{j1})^{-1}$ are of magnitude smaller than or equal to τ, while some coefficient of the partial fraction representation of $\prod_{j=1}^{k_1+1}(t - t_{j1})^{-1}$ is of magnitude larger than τ. Remove the poles $\{t_{j1}\}_{j=1}^{k_1}$ from the set T, and Leja order the remaining poles. Denote the ordered poles so obtained by $\{t_{j2}\}_{j=1}^{n-k_1}$. Next determine the partial fraction representation of the reciprocal polynomials $\prod_{j=1}^{k}(t - t_{j2})^{-1}$ for increasing values of k, until $k = n - k_1$ or until a coefficient of the partial fraction representation is of magnitude larger than τ. Continuing in this manner we obtain the IPF(τ) representation

$$\prod_{j=1}^{n} (t - t_j)^{-1} = \prod_{l=1}^{\mu} \left(\sum_{j=1}^{k_l} \frac{\alpha_{jl}}{t - t_{jl}} \right). \tag{18}$$

The organization of the computations required is described in the following algorithm.

Algorithm 3.8. Computation of an IPF(τ) representation (18) of a reciprocal monic polynomial.

Input: $T := \{t_j\}_{j=1}^n$, $\tau \geqslant 0$.

Output: $\cup_{l=1}^{\mu}\{t_{jl}\}_{j=1}^{k_l}$, $\cup_{l=1}^{\mu}\{\alpha_{jl}\}_{j=1}^{k_l}$, where $\sum_{l=1}^{\mu} k_l = n$.

$j := 1$; $\mu := 1$; $k := 0$;
while $j \leqslant n$ do
  $k := k + 1$;
  % $j - 1$ = total number of poles already selected
  % $k - 1$ = number of poles in present ($= \mu$th) factor of IPF(τ) representation
  if $k = 1$ then
    choose $t_{1\mu} \in T$ such that $|t_{1\mu}| = \min_{t\in T}|t|$;
    $\alpha_{1\mu} := 1$; $j := j + 1$;
  else

choose $t_{k\mu} \in T$ such that $\prod_{l=1}^{k-1} |t_{k\mu} - t_{l\mu}| = \max_{t \in T} \prod_{l=1}^{k-1} |t - t_{k\mu}|$;
for $l := 1, 2, \ldots, k-1$ do $\tilde{\alpha}_{l\mu} := \alpha_{l\mu}(t_{l\mu} - t_{k\mu})^{-1}$ endfor;
$\tilde{\alpha}_{k\mu} := \prod_{l=1}^{k-1} (t_{k\mu} - t_{l\mu})^{-1}$;
if $\max_{1 \leqslant l \leqslant k} |\tilde{\alpha}_{l\mu}| \leqslant \tau$ then
for $l := 1, 2, \ldots, k$ do $\alpha_{l\mu} := \tilde{\alpha}_{l\mu}$ endfor;
$j := j + 1$;
else
% begin new factor
$T := T \backslash \{t_{l\mu}\}_{l=1}^{k-1}$; $k_\mu := k - 1$; $k := 0$; $\mu := \mu + 1$;
endif
endif
endwhile

Generally, the smaller the value of τ, the larger the number of factors μ in the IPF representation. If $\tau = 0$, then Algorithm 3.8 yields the product form representation of $1/p$ with the poles ordered so that their magnitude increases with their index. This ordering is appropriate unless it causes overflow; see [36] for discussion on the ordering of the factors in a product form representation.

3.2.2. The case $m = n > 0$

We now turn to the determination of incomplete partial fraction representations of quotients of polynomials q and p of the same degree

$$p(t) = a_0 \prod_{j=1}^{n} (t - t_j), \qquad q(t) = b_0 \prod_{j=1}^{n} (t - s_j), \tag{19}$$

where we without loss of generality may assume that $b_0/a_0 = 1$. Theorem 4.3 of Section 4 presents a bound for the growth of the product of coefficients of the partial fraction representation of reciprocal polynomials, whose poles are Leja points. The proof of the theorem suggests the following ordering of the zeros s_j and poles t_j of q/p. Assume for the moment that the s_j and t_j are selected from two compact disjoint sets S and T in $\mathbb{C}$, such that each component of the complement $(\mathbb{C} \cup \{\infty\}) \backslash \{S \cup T\}$ is regular for the Dirichlet problem. Select s_1 and t_1 so that

$$|s_1 - t_1| = \inf_{\substack{s \in S \\ t \in T}} |s - t|, \quad s_1 \in S, \ t_1 \in T, \tag{20}$$

and let the points t_k and s_k for $k > 1$ satisfy

$$|s_k - t_k| \prod_{l=1}^{k-1} \frac{|t_k - s_l||s_k - t_l|}{|t_k - t_l|^2} = \inf_{\substack{s \in S \\ t \in T}} |s - t| \prod_{l=1}^{k-1} \frac{|t - s_l||s - t_l|}{|t - t_l|^2}, \quad s_k \in S, \ t_k \in T. \tag{21}$$

The points s_k and t_k are generally not determined uniquely by (20) and (21). We will show in Section 4 that a sequence $s_1, t_1, s_2, t_2, \ldots$ that satisfies (20) and (21) is an analogue of a sequence of Leja points.

We apply the ordering (20) and (21) to determine an incomplete partial fraction representation of q/p in the following manner. Let S be the set of zeros of q and T the set of zeros of p. Our requirements on p and q, stated after formula (1), secure that both the sets S and T consist of n distinct points, and that $S \cap T = \emptyset$. Order the elements of S and T according to (20) and (21). This

yields a sequence $s_{11}, s_{21}, s_{31}, \ldots$ of zeros of q and a sequence $t_{11}, t_{21}, t_{31}, \ldots$ of zeros of p. Compute the partial fraction representation of $\prod_{j=1}^{k}(t - s_{j1})/(t - t_{j1})$ from the partial fraction representation of $\prod_{j=1}^{k-1}(t - s_{j1})/(t - t_{j1})$ for increasing values of k. This process continues until $k = n$, or until a coefficient in the partial fraction representation is of magnitude larger than the given threshold τ. Let the integer $k_1 < n$ be such that the coefficients of the partial fraction representation of $\prod_{j=1}^{k_1}(t - s_{j1})/(t - t_{j1})$ are of magnitude smaller than or equal to τ, but a coefficient of the partial fraction representation of $\prod_{j=1}^{k_1+1}(t - s_{j1})/(t - t_{j1})$ is not. Remove the zeros $\{t_{j1}\}_{j=1}^{k_1}$ of p from the set T, and the zeros $\{s_{j1}\}_{j=1}^{k_1}$ of q from the set S, and then order the elements of the sets S and T so obtained according to (20) and (21). This yields a sequence $s_{12}, s_{22}, s_{32}, \ldots$ of zeros of q and a sequence of $t_{12}, t_{22}, t_{32}, \ldots$ of zeros of p. We proceed to determine the partial fraction representation of $\prod_{j=1}^{k}(t - s_{j2})/(t - t_{j2})$ for increasing values of k, until $k = n - k_1$ or until a coefficient in the partial fraction representation is of magnitude larger than τ. Continuing in this manner, we determine an IPF(τ) representation of the form (7). Details of the computations are described in the following algorithm.

Algorithm 3.9. Computation of an IPF(τ) representation (7) of the quotient of two polynomials (19) of equal degree, such that $b_0/a_0 = 1$.

Input: $T := \{t_j\}_{j=1}^{n}$, $S := \{s_j\}_{j=1}^{n}$, $\tau \geqslant 0$.

Output: $\bigcup_{l=1}^{\mu}\{t_{jl}\}_{j=1}^{k_l}$, $\bigcup_{l=1}^{\mu}\{\alpha_{jl}\}_{j=0}^{k_l}$, where $\sum_{l=1}^{\mu} k_l = n$.

$j := 1$; $\mu := 1$, $k := 0$;
while $j \leqslant n$ do
  $k := k + 1$;
  % $j - 1$ = total number of poles already selected
  % $k - 1$ = number of poles in present ($= \mu$th) factor of IPF(τ) representation
  if $k = 1$ then
    choose $s_{1\mu} \in S$ and $t_{1\mu} \in T$ such that $|s_{1\mu} - t_{1\mu}| = \min_{s\in S, t\in T}|s - t|$;
    $\alpha_{0\mu} := 1$; $\alpha_{1\mu} := t_{1\mu} - s_{1\mu}$; $j := j + 1$;
  else
    choose $s_{k\mu} \in S$ and $t_{k\mu} \in T$ such that
    $|s_{k\mu} - t_{k\mu}| \prod_{l=1}^{k-1}|t_{k\mu} - s_{l\mu}||s_{k\mu} - t_{l\mu}|/|t_{k\mu} - t_{l\mu}|^2$
      $= \min_{s\in S, t\in T}|s - t| \prod_{l=1}^{k-1}|t - s_{l\mu}||s - t_{l\mu}|/|t - t_{l\mu}|^2$,
    for $l := 1, 2, \ldots, k-1$ do $\tilde{\alpha}_{l\mu} := \alpha_{l\mu}(t_{l\mu} - s_{k\mu})/(t_{l\mu} - t_{k\mu})$ endfor;
    $\tilde{\alpha}_{k\mu} := (t_{k\mu} - s_{k\mu}) \prod_{l=1}^{k-1}(t_{k\mu} - s_{l\mu})/(t_{k\mu} - t_{l\mu})$,
    if $\max_{1\leqslant l\leqslant k}|\tilde{\alpha}_{l\mu}| \leqslant \tau$ then
      for $l := 1, 2, \ldots, k$ do $\alpha_{l\mu} := \tilde{\alpha}_{l\mu}$ endfor;
      $j := j + 1$;
    else
      % begin new factor
      $S := S \backslash \{s_{l\mu}\}_{l=1}^{k-1}$; $T := T \backslash \{t_{l\mu}\}_{l=1}^{k-1}$;
      $k_\mu := k - 1$; $k := 0$; $\mu := \mu + 1$;
    endif
  endif
endwhile

3.2.3. The case $0 < m < n$

We conclude this section with a modification of Algorithm 3.9 designed to determine an incomplete partial fraction representation with small coefficients of the quotient q/p of two polynomials p and q of different degrees. Let p be given by (19), and let q be defined by

$$q(t) := b_0 \prod_{j=1}^{m} (t - s_j), \quad b_0 \neq 0, \tag{22}$$

and assume that the degree m of q is at least one and strictly smaller than the degree n of p. Introduce the sets $S := \{s_j\}_{j=1}^{m}$ and $T := \{t_j\}_{j=1}^{n}$. For each element s_j we select from S, we select roughly n/m elements from T. Assume that we already have selected $l-1$ elements $s_{11}, s_{21}, \ldots, s_{l-1,1}$ from S and $k-1$ elements $t_{11}, t_{21}, \ldots, t_{k-1,1}$ from T, and that no coefficient of the partial fraction representation of

$$\frac{\prod_{j=1}^{l-1}(t - s_{j1})}{\prod_{j=1}^{k-1}(t - t_{j1})} \tag{23}$$

is of magnitude larger than a given threshold τ. If l/k approximates m/n as well as or better than $(l-1)/k$, i.e., if

$$\left| \frac{l}{k} - \frac{m}{n} \right| \leqslant \left| \frac{l-1}{k} - \frac{m}{n} \right|, \tag{24}$$

then we determine $s_l \in S$ and $t_k \in T$ such that

$$|t_k - s_l| \frac{\prod_{j=1}^{l-1} |t_k - s_j| \prod_{i=1}^{k-1} |t_i - s_l|}{\prod_{j=1}^{k-1} |t_k - t_j|^2} = \inf_{\substack{s \in S \\ t \in T}} |t - s| \frac{\prod_{j=1}^{l-1} |t - s_j| \prod_{i=1}^{k-1} |t_i - s|}{\prod_{j=1}^{k-1} |t - t_j|^2}. \tag{25}$$

Formula (25) is analogous to (21); see Section 4. We then determine the partial fraction representation of

$$\frac{\prod_{j=1}^{l}(t - s_{j1})}{\prod_{j=1}^{k}(t - t_{j1})} \tag{26}$$

from the partial fraction representation of (23).

On the other hand, if inequality (24) is violated, then we do not select an element of S, but only choose an element $t_k \in T$ such that

$$\frac{\prod_{j=1}^{l-1} |t_k - s_j|}{\prod_{j=1}^{k-1} |t_k - t_j|^2} = \inf_{t \in T} \frac{\prod_{j=1}^{l-1} |t - s_j|}{\prod_{j=1}^{k-1} |t - t_j|^2}. \tag{27}$$

Formula (27) is obtained by dropping the factors involving s_l in (25); see Section 4 for a motivation.

We then determine the partial fraction representation of

$$\frac{\prod_{j=1}^{l-1}(t-s_{j1})}{\prod_{j=1}^{k}(t-t_{j1})} \tag{28}$$

from the partial fraction representation of (23).

If a coefficient of the partial fraction representations of (26) or (28) is of magnitude larger than τ, then we remove the set $\{s_{j1}\}_{j=1}^{l-1}$ from S and the set $\{t_{j1}\}_{j=1}^{k-1}$ from T, and accept the partial fraction representation of (23) as a factor in our IPF(τ) representation of q/p. The next factor in the IPF(τ) representation is determined analogously from the remaining elements in S and T. If no coefficient of the partial fraction coefficient representations of (26) is of magnitude larger than τ, then we increase both k and l by 1, and, similarly, if no coefficient of the partial fraction representation of (28) is of magnitude larger than τ, then we increase k by one, but keep l fixed. We are now in the same position as when we considered the partial fraction representation of (23), and continue in the manner described above.

We have so far ignored that the decision whether to increase l also depends on the number of elements in the sets S and T that have not been selected yet. One has to make sure that throughout the computation of the IPF(τ) representation, the number of elements in T that have not been selected yet is at least as large as the number of elements in S that have not been selected yet. Details are presented in the following algorithm.

Algorithm 3.10. Compute IPF(τ) representation (7) of quotient of two polynomials q and p of different degrees, as given in (22) and (19), respectively, such that $b_0/a_0 = 1$.

Input: $T := \{t_j\}_{j=1}^{n}$, $S := \{s_j\}_{j=1}^{m}$, $\tau \geqslant 0$.

Output: $\bigcup_{l=1}^{\mu}\{t_{jl}\}_{j=1}^{k_l}$, $\bigcup_{l=1}^{\mu}\{\alpha_{jl}\}_{j=0}^{k_l}$, where $\sum_{l=1}^{\mu} k_l = n$.

$k := 1$; $\mu := 1$; $\hat{k} := 0$; $\hat{l} := 0$; $l := 0$; $\theta := m/n$;

while $k \leqslant n$ do

$\hat{k} := \hat{k} + 1$;

% $k-1$ = total number of poles already selected

% l = total number of zeros already selected

% $\hat{k}-1$ = number of poles in present ($= \mu$th) factor of IPF(τ) representation

% $\hat{l}$ = number of zeros in present ($= \mu$th) factor of IPF(τ) representation

if $|l+1-k\theta| > |l-k\theta|$ and $n-k+1 > m-l$ then

% l and $\hat{l}$ are not increased

if $\hat{k} = 1$ then

choose $t_{1\mu} \in T$ such that $|t_{1\mu}| = \min_{t\in T}|t|$;

$\alpha_{1\mu} := 1$; $k := k+1$;

else

choose $t_{\hat{k}\mu} \in T$ such that

$$\frac{\prod_{j=1}^{\hat{l}}|t_{\hat{k}\mu}-s_{j\mu}|}{\prod_{j=1}^{\hat{k}-1}|t_{\hat{k}\mu}-t_{j\mu}|^2} = \min_{t\in T}\frac{\prod_{j=1}^{\hat{l}}|t-s_{j\mu}|}{\prod_{j=1}^{\hat{k}-1}|t-t_{j\mu}|^2};$$

for $j := 1, 2, \ldots, \hat{k}-1$ do $\tilde{\alpha}_{j\mu} := \alpha_{j\mu}(t_{j\mu} - t_{\hat{k}\mu})^{-1}$ endfor;

$\tilde{\alpha}_{\hat{k}\mu} := \prod_{j=1}^{\hat{l}} (t_{\hat{k}\mu} - s_{j\mu}) / \prod_{j=1}^{\hat{k}-1} (t_{\hat{k}\mu} - t_{j\mu})$;

if $\max_{1 \leqslant j \leqslant \hat{k}} |\tilde{\alpha}_{j\mu}| \leqslant \tau$ then
for $j := 1, 2, \ldots, \hat{k}$ do $\alpha_{j\mu} := \tilde{\alpha}_{j\mu}$;
$k := k + 1$;
else
% begin new factor
if $\hat{k} - 1 = \hat{l}$ then $\alpha_{0\mu} := 1$ else $\alpha_{0\mu} := 0$ endif;
$S := S \setminus \{s_{j\mu}\}_{j=1}^{\hat{l}}$; $T := T \setminus \{t_{j\mu}\}_{j=1}^{\hat{k}-1}$;
$\hat{k} := \hat{k} - 1$; $\mu := \mu + 1$; $\hat{k} := 0$; $\hat{l} := 0$;
endif
endif
else
% l and $\hat{l}$ are increased if partial fraction coefficients sufficiently small
$\hat{l} := \hat{l} + 1$;
if $\hat{k} = 1$ and $\hat{l} = 1$ then
choose $s_{1\mu} \in S$ and $t_{1\mu} \in T$ such that $|s_{1\mu} - t_{1\mu}| = \min_{s \in S, t \in T} |s - t|$;
$\alpha_{1\mu} := t_{1\mu} - s_{1\mu}$; $k := k + 1$; $l := l + 1$;
else
choose $s_{\hat{l}\mu} \in S$ and $t_{\hat{k}\mu} \in T$ such that

$$|t_{\hat{k}\mu} - s_{\hat{l}\mu}| \frac{\prod_{j=1}^{\hat{l}-1} |t_{\hat{k}\mu} - s_{j\mu}| \prod_{j=1}^{\hat{k}-1} |t_{j\mu} - s_{\hat{l}\mu}|}{\prod_{j=1}^{\hat{k}-1} |t_{\hat{k}\mu} - t_{j\mu}|^2}$$

$$= \min_{\substack{s \in S \\ t \in T}} |t - s| \frac{\prod_{j=1}^{\hat{l}-1} |t - s_{j\mu}| \prod_{j=1}^{\hat{k}-1} |t_{j\mu} - s|}{\prod_{j=1}^{\hat{k}-1} |t - t_{j\mu}|^2};$$

for $j := 1, 2, \ldots, \hat{k} - 1$ do $\tilde{\alpha}_{j\mu} := \alpha_{j\mu}(t_{j\mu} - s_{\hat{l}\mu}) / (t_{j\mu} - t_{\hat{k}\mu})$ endfor;
$\tilde{\alpha}_{\hat{k}\mu} := \prod_{j=1}^{\hat{l}} (t_{\hat{k}\mu} - s_{j\mu}) / \prod_{j=1}^{\hat{k}-1} (t_{\hat{k}\mu} - t_{j\mu})$;
if $\max_{1 \leqslant j \leqslant \hat{k}} |\tilde{\alpha}_{j\mu}| \leqslant \tau$ then
for $j := 1, 2, \ldots, \hat{k}$ do $\alpha_{j\mu} := \tilde{\alpha}_{j\mu}$ endfor;
$k := k + 1$; $l := l + 1$;
else
% begin new factor
if $\hat{k} = \hat{l}$ then $\alpha_{0\mu} := 1$ else $\alpha_{0\mu} := 0$ endif;
$S := S \setminus \{s_{j\mu}\}_{j=1}^{\hat{l}-1}$; $T := T \setminus \{t_{j\mu}\}_{j=1}^{\hat{k}-1}$;
$k_\mu := \hat{k} - 1$; $\mu := \mu + 1$; $\hat{k} := 0$; $\hat{l} := 0$;
endif
endif
endif
endwhile

Some properties of the incomplete partial fraction representations determined by Algorithms 3.8–3.10 are shown in the next section.

4. Properties of coefficients of partial fraction representations

Let T be a compact set in $\mathbb{C}$, such that its complement $(\mathbb{C}\cup\{\infty\})\setminus T$ is connected and regular for the Dirichlet problem. Introduce the sequence of monic polynomials

$$p_n(t) := \prod_{j=1}^{n} (t - t_j^{(n)}), \quad t_j^{(n)} \in T, \ n = 1, 2, \ldots, \tag{29}$$

and assume that $t_j^{(n)} \neq t_k^{(n)}$ for $j \neq k$. We can bound the coefficients $\alpha_j^{(n)}$ in the partial fraction representation

$$\frac{1}{p_n(t)} = \sum_{j=1}^{n} \frac{\alpha_j^{(n)}}{t - t_j^{(n)}}, \qquad \alpha_j^{(n)} := \prod_{\substack{l=1 \\ l\neq j}}^{n} (t_j^{(n)} - t_l^{(n)})^{-1}, \tag{30}$$

for certain special distributions of the poles $t_j^{(n)}$ of $1/p_n$.

Example 4.1. Let γ be a positive constant and define $T := \{z: |z| \leqslant \gamma\}$. Let $t_j^{(n)} := \gamma \exp(2\pi\imath(j-1)/n)$ for $1 \leqslant j \leqslant n$. Then $p_n(t) = t^n - \gamma^n$ and, therefore, $\alpha_j^{(n)} = 1/p_n'(t_j^{(n)}) = n^{-1}(t_j^{(n)})^{1-n}$. Thus, $|\alpha_j^{(n)}| = n^{-1}\gamma^{1-n}$ for $1 \leqslant j \leqslant n$. We note that if $n = 2^k$ for some integer $k \geqslant 0$, then the poles $t_j^{(n)}$, $1 \leqslant j \leqslant n$, are Leja points for T; cf. Example 3.6.

Example 4.2. Let T and γ be as in Example 4.1 and define $t_j^{(n)} := \gamma t_j$ for $1 \leqslant j \leqslant n$, where the t_j are given by (17). It follows from [13, Lemma 2.3] that $|\alpha_j^{(n)}| \leqslant n\gamma^{1-n}$ for $1 \leqslant j \leqslant n$. We note that the $t_j^{(n)}$ are Leja points for T; cf. Example 3.6.

It is difficult to bound the magnitude of the coefficients of $\alpha_j^{(n)}$ if the poles $t_j^{(n)}$ of the partial fraction (30) are Leja points for general sets T. However, it is fairly straightforward to bound the growth of the products $\prod_{j=1}^{n} |\alpha_j^{(n)}|$ with n.

Theorem 4.3. *Assume that the set T satisfies the conditions stated in the beginning of this section and let $\{t_j^{(n)}\}_{j=1}^{n}$ be the first n points of a sequence of Leja points for T. Let the coefficients $\{\alpha_j^{(n)}\}_{j=1}^{(n)}$ be defined by* (30). *Let $\{\tau_j^{(n)}\}_{j=1}^{n}$ be pairwise distinct, but otherwise arbitrary, points in T and let $\{\tilde{\alpha}_j^{(n)}\}_{j=1}^{n}$ denote the coefficients of the partial fraction representation of $\prod_{j=1}^{n}(t - \tau_j^{(n)})^{-1}$. Then*

$$\prod_{j=1}^{n} |\alpha_j^{(n)}| \leqslant \chi^{-n(n-1)} \tag{31}$$

and

$$\lim_{n\to\infty} \prod_{j=1}^{n} |\alpha_j^{(n)}|^{1/(n(n-1))} = \chi^{-1} \leqslant \liminf_{n\to\infty} \prod_{j=1}^{n} |\tilde{\alpha}_j^{(n)}|^{1/(n(n-1))}, \tag{32}$$

where the constant $\chi = \chi(T)$ is the transfinite diameter of T.

Remark 4.4. Properties of the transfinite diameter are discussed in [31, 52]. Sometimes the transfinite diameter of a set T is referred to as the capacity of T.

Proof of Theorem 4.3. It follows from (30) that

$$\prod_{j=1}^{n} |\alpha_j^{(n)}| = \prod_{j=1}^{n} \prod_{\substack{l=1 \\ l \neq j}}^{n} |t_j^{(n)} - t_l^{(n)}|^{-1} = \prod_{j=2}^{n} \prod_{l=1}^{j-1} |t_j^{(n)} - t_l^{(n)}|^{-2}. \tag{33}$$

The fact that the $t_j^{(n)}$ are Leja points for T yields

$$\prod_{l=1}^{j-1} |t_j^{(n)} - t_l^{(n)}| \geqslant \chi^{j-1}, \tag{34}$$

where $\chi = \chi(T)$ is the transfinite diameter of T; see [31]. Combining (33) and (34) shows (31).

We turn to the proof of (32). Let the points $\{\hat{t}_j^{(n)}\}_{j=1}^{n}$ be Fekete points for T, i.e., they satisfy $\hat{t}_j \in T$ for all j and

$$\prod_{j=1}^{n} \prod_{\substack{l=1 \\ l \neq j}}^{n} |\hat{t}_j^{(n)} - \hat{t}_l^{(n)}| \geqslant \max_{\tilde{t}_k \in T} \prod_{j=1}^{n} \prod_{\substack{l=1 \\ l \neq j}}^{n} |\tilde{t}_j^{(n)} - \tilde{t}_l^{(n)}|. \tag{35}$$

Properties of Fekete points are discussed in, e.g., [15, 52], where it is shown that

$$\lim_{n \to \infty} \prod_{j=1}^{n} \prod_{\substack{l=1 \\ l \neq j}}^{n} |\hat{t}_j^{(n)} - \hat{t}_l^{(n)}|^{1/(n(n-1))} = \chi. \tag{36}$$

In view of that

$$\hat{\alpha}_j^{(n)} = \prod_{\substack{l=1 \\ l \neq j}}^{n} (\tau_j^{(n)} - \tau_l^{(n)})^{-1},$$

we obtain from (35) that

$$\prod_{j=1}^{n} |\hat{\alpha}_j^{(n)}|^{-1} = \prod_{j=1}^{n} \prod_{\substack{l=1 \\ l \neq j}}^{n} |\tau_j^{(n)} - \tau_l^{(n)}| \leqslant \prod_{j=1}^{n} \prod_{\substack{l=1 \\ l \neq j}}^{n} |\hat{t}_j^{(n)} - \hat{t}_l^{(n)}|. \tag{37}$$

It now follows from (37) and (36) that

$$\limsup_{n \to \infty} \prod_{j=1}^{n} |\hat{\alpha}_j^{(n)}|^{-1/(n(n-1))} \leqslant \chi.$$

Therefore,

$$\liminf_{n \to \infty} \prod_{j=1}^{n} |\hat{\alpha}_j^{(n)}|^{1/(n(n-1))} \leqslant \chi^{-1}. \tag{38}$$

Finally, the sequence of products $\prod_{j=1}^{n} |\alpha_j^{(n)}|^{1/(n(n-1))}$, $n = 1, 2, \ldots$, satisfies both (31) and (38). This completes the proof of the theorem. □

Consider the polynomial p_{n+1}, defined by (29), and assume that $t_j^{(n+1)} = t_j^{(n)}$ for $1 \leqslant j \leqslant n$. Then the coefficients $\alpha_j^{(n+1)}$ in the partial fraction representation of $1/p_{n+1}$ satisfy

$$\prod_{j=1}^{n+1} |\alpha_j^{(n+1)}| = \prod_{j=1}^{n} |\alpha_j^{(n)}| \cdot \prod_{j=1}^{n} |t_{n+1}^{(n+1)} - t_j^{(n+1)}|^{-2}, \tag{39}$$

where the $\alpha_j^{(n)}$ are given by (30). The proof of Theorem 4.3 is based on the fact that $t_{n+1}^{(n+1)} \in T$ is chosen so that

$$\prod_{j=1}^{n} |t_{n+1}^{(n+1)} - t_j^{(n+1)}|^{-2} = \inf_{t \in T} \prod_{j=1}^{n} |t - t_j^{(n+1)}|^{-2}. \tag{40}$$

Algorithms 3.9 and 3.10 are modifications of Algorithm 3.8 that allow the determination of IPF(τ) representations of rational functions q/p, where q is a polynomial of degree not larger than the degree of p. In order to motivate their design, let S and T be two disjoint compact sets in $\mathbb{C}$, and assume that they satisfy suitable regularity conditions. Let p_n be given by (29) and define

$$q_n(t) := \prod_{j=1}^{n} (s - s_j^{(n)}), \quad s_j^{(n)} \in S. \tag{41}$$

Then

$$\frac{q_n(t)}{p_n(t)} = 1 + \sum_{j=1}^{n} \frac{\alpha_j^{(n)}}{t - t_j^{(n)}}, \tag{42}$$

where the coefficients of $\alpha_j^{(n)}$ satisfy

$$\prod_{j=1}^{n} |\alpha_j^{(n)}| = \prod_{j=1}^{n} \prod_{l=1}^{n} |t_j^{(n)} - s_l^{(n)}| \Big/ \prod_{j=1}^{n} \prod_{\substack{l=1 \\ l \neq j}}^{n} |t_j^{(n)} - t_l^{(n)}|. \tag{43}$$

Let the zeros $t_j^{(n+1)}$ of the polynomial p_{n+1}, given by (29), satisfy $t_j^{(n+1)} = t_j^{(n)}$ for $1 \leqslant j \leqslant n$. Similarly, let the zeros $s_j^{(n+1)}$ of the polynomial q_{n+1}, given by (41), satisfy $s_j^{(n+1)} = s_j^{(n)}$ for $1 \leqslant j \leqslant n$. Let $\alpha_j^{(n+1)}$ be the coefficients of the partial fraction representation of q_{n+1}/p_{n+1}; cf. (42). Then (43) yields a formula analogous to (39),

$$\prod_{j=1}^{n+1} |\alpha_j^{(n+1)}| = \prod_{j=1}^{n} |\alpha_j^{(n)}| \cdot |t_{n+1}^{(n+1)} - s_{n+1}^{(n+1)}| \prod_{j=1}^{n} \frac{|t_{n+1}^{(n+1)} - s_j^{(n+1)}||s_{n+1}^{(n+1)} - t_j^{(n+1)}|}{|t_{n+1}^{(n+1)} - t_j^{(n+1)}|^2}.$$

Formula (40) suggests that we choose $t_{n+1}^{(n+1)} \in T$ and $s_{n+1}^{(n+1)} \in S$ such that

$$\begin{aligned} &|t_{n+1}^{(n+1)} - s_{n+1}^{(n+1)}| \prod_{j=1}^{n} \frac{|t_{n+1}^{(n+1)} - s_j^{(n+1)}||s_{n+1}^{(n+1)} - t_j^{(n+1)}|}{|t_{n+1}^{(n+1)} - t_j^{(n+1)}|^2} \\ &\quad = \inf_{\substack{s \in S \\ t \in T}} |t - s| \prod_{j=1}^{n} \frac{|t - s_j^{(n+1)}||s - t_j^{(n+1)}|}{|t - t_j^{(n+1)}|^2}. \end{aligned} \tag{44}$$

Thus, we have derived formula (21). Note that when $k = 1$ in (21), then formula (20) is obtained. We are presently studying a generalization of Leja points suggested by the minimization problem (44).

Computed examples in Section 5 indicate that this generalization is appropriate for determining incomplete partial fraction representations with coefficients of small magnitude. We remark that other generalizations of Leja points are investigated in [2, 32].

The minimization problems (25) and (27) can be derived similarly as (44) by considering the product of the coefficients of the partial fraction representations q_l/p_n, q_{l+1}/p_{n+1} and q_l/p_{n+1}, where p_j is given by (29) and q_j by (41). This motivates our selection of zeros and poles in Algorithm 3.10.

5. Numerical experiments

In this section we present numerical examples illustrating how the algorithms described in Section 3 can be used to control the size of the coefficients of the partial fraction representations of the rational function and hence avoid the associated stability problems. We first present examples from the evaluation of rational functions of the form (1) with scalar argument. A few of these examples have previously been reported in [7].

5.1. The scalar case

In the examples that follow, we examine the error in the computation of some rational functions using IPF(τ) representations for several values of τ. We evaluate these expressions for the rational function at some value t, and compare the result with that obtained when we express the rational function as the ratio of two polynomials in product form, using the IPF(0) representation. Numerical tests in this section were carried out on an IBM RISC 6000/550 workstation using 64-bit arithmetic, i.e., approximately 15 significant decimal digits. Whenever random numbers are used, they are chosen from a uniform distribution in the specified interval.

Table 1 shows the poles and zeros for the rational functions used in the experiments of this section, and Table 2 summarizes the results. The first three columns of Table 2 define the rational function to be evaluated. In particular, they show the numerator degree m, the denominator degree n and the point t at which the rational function is to be evaluated. The fourth column shows the value of τ used to generate the IPF(τ) representation, and column five displays the number of factors μ in the IPF(τ) representations determined by Algorithms 3.8–3.10. The last column shows

Table 1
Roots and poles for the rational functions used in the scalar experiments

Type	Description
1	t_j random in $[-1, 1]$
2	$t_j := j/n$
3	s_j random in $[-2, 0]$; $t_j := j/n$
4	s_j random in $[0, 1]$; $t_j := j/n$

Table 2
Experimental results from the use of IPF(τ) representations to evaluate scalar rational functions

Type	(m,n)	t	τ	Factors	Value
1	(0, 40)	1.5	0	40	$0.58640286 \cdot 10^{-5}$
			$1 \cdot 10^{5}$	4	$0.58640286 \cdot 10^{-5}$
			$1 \cdot 10^{7}$	3	$0.58645875 \cdot 10^{-5}$
			$1 \cdot 10^{9}$	2	$0.58699068 \cdot 10^{-5}$
			∞	1	0.50000000
2	(0, 40)	2.2	0	40	$0.148 \cdot 10^{-8}$
			$1 \cdot 10^{10}$	3	$0.152 \cdot 10^{-8}$
			∞	1	*
3	(20, 20)	−3	0	20	$0.34323635 \cdot 10^{-6}$
			$1 \cdot 10^{7}$	3	$0.34323558 \cdot 10^{-6}$
			$1 \cdot 10^{8}$	3	$0.34323262 \cdot 10^{-6}$
			$1 \cdot 10^{9}$	2	$0.34320468 \cdot 10^{-6}$
			∞	1	*
4	(10, 20)	3.5	0	20	$0.1423 \cdot 10^{-2}$
			$1 \cdot 10^{13}$	2	$0.1550 \cdot 10^{-2}$
			∞	1	*
4	(10, 20)	7.5	0	20	$0.2940 \cdot 10^{-7}$
			$1 \cdot 10^{8}$	2	$0.2939 \cdot 10^{-7}$
			$1 \cdot 10^{13}$	2	$0.2373 \cdot 10^{-4}$
			∞	1	*

* means that no correct significant digits were computed.

the value obtained by evaluating the IPF(τ) representation at the point t. An asterisk indicates that the value obtained by evaluating the IPF(τ) representation did not have any correct digit.

Example 5.1. Consider the rational function of type 1, $r(t) := 1/p(t)$, where $p(t) := \prod_{k=1}^{40}(t - t_k)$. If we evaluate $r(t)$ at $t = 1.5$ using its partial fraction representation, then we lose all significant digits. Table 2 shows the computed values obtained with IPF(τ) representations at $t = 1.5$ for several values of τ.

Example 5.2. Consider the reciprocal polynomial $r(t) = \prod_{k=1}^{40}(t - k/40)^{-1}$. Rational functions of this form were discussed in Example 3.3. The coefficients of the (complete) partial fraction decomposition of this function become very large. For instance, the coefficient for $(t - 1/40)^{-1}$ is of magnitude $6 \cdot 10^{15}$. This suggests that the use of the complete partial fraction decomposition may lead to severe loss of significant digits. We expect the accuracy achieved in the rational function evaluation when using a complete partial fraction representation to depend on the exact value of r at the point t, the larger values of $r(t)$ being less sensitive to the presence of large coefficients. We evaluated the rational function at $t = 2.2$. Table 2 shows that if we use the complete partial fraction

representation, then we lose all significant digits. Already for $\tau = 1 \cdot 10^{10}$, the IPF(τ) representation yields 2 significant digits in the computed value of $r(2.2)$.

Example 5.3. Let

$$r(t) := \frac{\prod_{k=1}^{20}(t - s_k)}{\prod_{j=1}^{20}(t - t_j)}$$

be a rational function of type 3. When using the complete partial fraction decomposition to evaluate $r(-3)$, we obtained no significant digits. Decreasing the value of τ gave increased accuracy in the computed value of $r(-3)$.

Example 5.4. The rational function of type 4 considered in this example is given by

$$r(t) := \frac{\prod_{k=1}^{10}(t - s_k)}{\prod_{k=1}^{20}(t - k/20)}.$$

The presence of large coefficients in the IPF(τ) representation of r does not affect the accuracy of the result as much when $|r(t)|$ is large as when $|r(t)|$ is small. Table 2 lists the results obtained using IPF(τ) representations for different values of τ to evaluate r at $t = 3.5$ and $t = 7.5$.

The numerical experiments show that when the IPF(τ) representation contains coefficients of large magnitude, the accuracy of the computed value when evaluating the IPF(τ) representation may be low. On the other hand, when τ is sufficiently small, the IPF(τ) representation yields accurate values. The examples indicate that often much higher accuracy can be achieved with only few factors in the IPF(τ) representation than when the complete partial fraction representation is used. Thus, not much parallelism is lost in the quest for accuracy.

Consider the partial fraction expansion (2) of $r = q/p$. For a fixed value of t, an estimate of the number of decimal digits lost by using the partial fraction representation of r is given by the formula

$$\text{digits lost} := \log_{10}\left(\max\left\{|\alpha_0|, \frac{|\alpha_1|}{|t - t_1|}, \ldots, \frac{|\alpha_n|}{|t - t_n|}\right\}/|r(t)|\right). \tag{45}$$

If we assume that $|t - t_j| \geqslant 1$ for all $1 \leqslant j \leqslant n$, then (45) yields the bound

$$\text{digits lost} \leqslant \log_{10}(\max\{|\alpha_0|, |\alpha_1|, \ldots, |\alpha_n|\}/|r(t)|),$$

from which we obtain

$$\text{digits lost} \leqslant \log_{10}(\tau/|r(t)|). \tag{46}$$

The number of desired significant (decimal) digits in the computed value of $r(t)$ is the number of significant digits in the computer arithmetic minus the number of digits lost. This observation

yields the following ad hoc rule for the choice of τ in order to obtain an IPF(τ) representation that yields a desired number of correct significant digits:

$$\begin{aligned}\log_{10}(\tau) = &\text{(number of significant digits in the computer arithmetic)}\\ &+ \log_{10}|r(t)|\\ &- \text{(number of desired correct significant digits in the value of the incomplete}\\ &\text{partial fraction),}\end{aligned} \tag{47}$$

where $r(t)$ denotes the exact value of the rational function to be evaluated at the point t. Thus, this choice of τ depends on the value of r at t.

5.2. The matrix case

This section discusses the application of IPF(τ) representations to the evaluation of matrix rational functions. We demonstrate the effectiveness of incomplete partial fraction representations in high-order integration methods for differential equations. These methods require the evaluation of high-degree rational approximants of the matrix exponential times a vector, i.e., one has to evaluate rational approximants of $e^{-A\delta}b$, where A is a large, possibly sparse, matrix; b is a vector; and $\delta > 0$ is a scalar. These approximants are of the form (1), where the rational function $r = q/p$ is chosen to approximate the exponential on a region $\Omega \subset \mathbb{C}$, which should contain the spectrum of the matrix $-A\delta$. Low-order approximants (of an order between 1 and 4) are used extensively in the literature. Parallel computation makes rational approximants of higher order of interest; see [20, 30, 39]. We show in this section that IPF(τ) representations, determined by the algorithms of Section 3, of rational approximants of the exponential function can be useful for the implementation of integration methods of high order. Throughout the section we assume that conditions justify the use of high-order integration methods.

Experiments reported in this section were carried out on an Alliant FX/2800 computer using 10 of 20 processors and running the Concentrix 3.0 operating system. Codes were written in Fortran using 64-bit arithmetic and compiled with the $-O$ option, so that vectorization and parallelization are carried out by the compiler. Timings were obtained by using the Alliant library function etime.

Implicit integration methods based on rational Padé approximants are quite popular due to their favorable stability and approximation properties. Denote the $[m/n]$ Padé approximant to e^t by

$$\tau_{m,n}(t) = \frac{q_m(t)}{p_n(t)},$$

where m and n denote the degrees of the numerator and denominator polynomials, respectively. Formulas for these polynomials are explicitly available, and it is known that all the zeros and poles of $r_{m,n}$ are simple; see [22, 42–44, 53, 54, 62]. The approximation error satisfies

$$|e^t - r_{m,n}(t)| = \begin{cases} \dfrac{m!n!}{(m+n)!(m+n+1)!}|t|^{m+n+1} + O(|t|^{m+n+2}) & \text{as } |t| \to 0,\\ O(|t|^{m-n}) & \text{as } |t| \to \infty. \end{cases} \tag{48}$$

We consider A-acceptable integration methods because of their practical importance. For these methods the rational approximants satisfy

$$|r_{m,n}(t)| < 1, \qquad \mathrm{Re}\, t < 0. \tag{49}$$

This condition only holds when $n-2 \leqslant m \leqslant n$; see [58]. In view of this, we primarily consider $[n/n]$ (diagonal) Padé approximants because they offer the highest order of approximation, but we note that for certain problems $[(n-1)/n]$ Padé approximants may offer certain advantages. We observe that the A-acceptability of diagonal Padé approximants was demonstrated by Birkhoff and Varga in [4].

A careful examination of formula (48) shows that when n is increased, the error at a fixed point t is reduced, and, moreover, the region around the origin in which $r_{n,n}$ gives an acceptable approximation of e^t is enlarged. The latter property of $r_{n,n}$ allows the use of larger time steps in the integration formula when n is large. The use of high-order methods, i.e., methods based on rational approximants $r_{m,n}$ with m or n large, is particularly attractive in a multiprocessor environment when an incomplete partial fraction representation of $r_{m,n}$ can be evaluated in parallel. We illustrate properties of high-order methods and associated partial fraction and incomplete partial fraction representations in a sequence of examples.

Example 5.5. Let $\rho^{\max}_{(m,n)}$ denote the magnitude of the coefficient of largest magnitude in the partial fraction representation of the $[m/n]$ Padé approximant of e^t. Table 3 shows $\rho^{\max}_{(m,n)}$ for several values of m and n and illustrates that the partial fraction coefficients can be of very large magnitude. For instance, $\rho^{\max}_{(10,10)} > 1 \cdot 10^5$ and $\rho^{\max}_{(20,20)} > 1 \cdot 10^{11}$. Thus, A-acceptable Padé approximants can have very large partial fraction coefficients. This motivates the use of incomplete partial fraction representations for their evaluation.

In the previous example, as well as in the examples below, we computed the zeros of the numerator and denominator polynomials q_m and p_n by first transforming each polynomial into its companion matrix representation, and then using subroutines from the LAPACK library [1] to compute the eigenvalues of the companion matrices. This method of computing zeros was chosen for its simplicity, but the question of how to accurately determine zeros of high-degree polynomials

Table 3
$\log_{10} \rho^{\max}_{(m,n)}$: base-10 logarithm of the magnitude of the partial fraction coefficient of largest magnitude of $[m/n]$ Padé approximants of e^t

n	$m=2$	$m=4$	$m=8$	$m=10$	$m=14$	$m=18$	$m=20$	$m=40$
2	1.1	—	—	—	—	—	—	—
4	1.3	2.3	—	—	—	—	—	—
8	1.8	2.7	4.6	—	—	—	—	—
10	2.0	3.0	4.8	5.8	—	—	—	—
14	2.5	3.5	5.3	6.3	8.1	—	—	—
18	3.0	4.0	5.9	6.8	8.6	10.4	—	—
20	3.3	4.2	6.1	7.0	8.8	10.6	11.5	—
40	5.7	6.7	8.7	9.6	11.7	11.3	12.2	17.6

deserves further attention. When the zeros of the numerator and denominator polynomials are explicitly known, the partial fraction coefficients are readily computed from formula (3).

The large partial fraction coefficients of Example 5.5 suggests that partial fraction representations of Padé approximants of $e^{-A\delta}$ may yield low accuracy due to cancellation of significant digits. The following example shows that this is indeed the case.

Example 5.6. Introduce the tridiagonal matrix $A := (1/h^2)\ \mathrm{trid}[-1\ \ 2\ \ -1]$ that arises from the discretization of the one-dimensional Laplace operator with mesh size $h := 1/(N+1)$ and $N := 998$ grid points. The eigenvalues

$$\lambda_j := \frac{4}{h^2}\sin^2\frac{j\pi}{2(N+1)} \quad (1 \leqslant j \leqslant N)$$

as well as associated eigenvectors v_j of A are explicitly known. Assume that the eigenvectors are of unit length and have positive first component, and define the vector

$$b := \sum_{j=1}^{N} \frac{1}{j} v_j. \tag{50}$$

Because b contains components of eigenvectors associated with large eigenvalues of A, a small time step δ is required in order to achieve acceptable accuracy. Table 4 shows the maximum relative error when using $[n/n]$ Padé approximants for $\exp(-\delta A)$, for several values of n and δ. The table shows that in order to achieve acceptable accuracy for large time steps δ, fairly large values of n are required. The linear systems of equations were solved by a direct tridiagonal solver.

We used the IPF(0) representation of the Padé approximants for the computations for Table 4. The corresponding results when the IPF(∞) representation of the Padé approximants are used are shown in Table 5. A comparison between Tables 4 and 5 shows that the IPF(∞) representations cause significant accuracy degradation for large values of n.

The following example illustrates that the significant loss of accuracy caused by the partial fraction representation can be avoided by using an incomplete partial fraction representation.

Table 4
$\log_{10}$ of maximum relative error of $e^{-A\delta}b$, based on $[n/n]$ Padé approximant of $e^{-A\delta}$, and using the IPF(0) representation for various values of δ and n

δ	$n=4$	$n=10$	$n=14$	$n=18$	$n=20$	$n=28$
$1\cdot10^{-5}$	−2	−4	−6	−9	−10	−13
$2\cdot10^{-5}$	−1	−2	−4	−5	−6	−8
$3\cdot10^{-5}$	−1	−2	−3	−4	−5	−6
$4\cdot10^{-5}$	−1	−2	−2	−3	−4	−5

Table 5
$\log_{10}$ of maximum relative error of $e^{-A\delta}b$, based on $[n/n]$ Padé approximant of $e^{-A\delta}$, and using the IPF(∞) representation for various values of δ and n

δ	$n=4$	$n=10$	$n=14$	$n=18$	$n=20$	$n=28$
$1\cdot10^{-5}$	−2	−4	−6	−6	−6	−2
$2\cdot10^{-5}$	−1	−2	−4	−5	−6	−2
$3\cdot10^{-5}$	−1	−2	−3	−4	−5	−2
$4\cdot10^{-5}$	−1	−2	−2	−3	−4	−2

Table 6
Components and $\log_{10}$ of maximum relative errors where $e^{-A\delta}b$ is evaluated based on the $[n/n]$ Padé approximant of $e^{-A\delta}$, and using the IPF(τ) representation, for $\delta = 1\cdot10^{-5}$ and $\delta = 4\cdot10^{-5}$

n	τ	Components	Error for δ		n	τ	Components	Error for δ	
			$1\cdot10^{-5}$	$4\cdot10^{-5}$				$1\cdot10^{-5}$	$4\cdot10^{-5}$
4	0	{1,...,1}	−1.0	−0.4	20	0	{1,...,1}	−9.7	−3.2
	10^4	{4}	−1.0	−0.4		10^4	{8,5,4,3}	−9.7	−3.2
	10^8	{4}	−1.0	−0.4		10^8	{15,5}	−9.2	−3.2
	∞	{4}	−1.0	−0.4		∞	{20}	−4.8	−3.2
14	0	{1,...,1}	−5.5	−1.9	24	0	{1,...,1}	−13.1	−4.3
	10^4	{8,4,2}	−5.5	−1.9		10^4	{9,5,5,3,2}	−12.7	−4.3
	10^8	{13,1}	−5.5	−1.9		10^8	{16,8}	−9.1	−4.3
	∞	{14}	−5.5	−1.9		∞	{24}	−2.6	−2.5
18	0	{1,...,1}	−8.2	−2.7	28	0	{1,...,1}	−13.7	−5.5
	10^4	{8,5,4,1}	−8.2	−2.7		10^4	{8,6,5,5,3,1}	−12.6	−5.5
	10^8	{15,3}	−8.2	−2.7		10^8	{18,9,1}	−8.6	−5.5
	∞	{18}	−6.1	−2.7		∞	{28}	−1.7	−1.6

Example 5.7. Let $r_{n,n}$ denote an IPF(τ) representation of the $[n/n]$ Padé approximants of e^t determined by Algorithm 3.9, and let the matrix A and vector b be the same as in Example 5.6. Table 6 shows the relative error $\|r_{n,n}(-A\delta)b - e^{-A\delta}b\|_\infty / \|e^{-A\delta}b\|_\infty$ for several values of n, τ and δ. Here $\|\cdot\|_\infty$ denotes the uniform vector norm. The columns labeled "Components" show the number of poles in each factor of the IPF(τ) representations, i.e., the columns show the values of $k_1, k_2, \ldots, k_\mu$ in formula (7). The columns labeled "Error" show the base 10 logarithm of the relative error in the approximation for each of the time steps δ.

When the rational function $r_{m,n}$ has real coefficients, the complex poles and coefficients in its partial fraction representation appear in complex conjugate pairs. This can be exploited to reduce the number of linear systems of equations that need be solved to evaluate $r_{m,n}(-A\delta)b$ when $A \in \mathbb{R}^{N\times N}$, $b \in \mathbb{R}^N$ and $\delta > 0$ in the following manner [18, 29, 63]. Let t_j be a pole of $r_{m,n}$ with

nonvanishing imaginary part, and let α_j be the partial fraction coefficient associated with t_j. Let $t_{j+1} := \bar{t}_j$ and $\alpha_{j+1} := \bar{\alpha}_j$, where the bar denotes complex conjugation. Then the evaluation of $r_{m,n}(-A\delta)b$ requires the computation of $x_j := \alpha_j(-A\delta - t_j I)^{-1}b$ and $x_{j+1} := \alpha_{j+1}(-A\delta - t_{j+1}I)^{-1}b$. This can be done by solving only one linear system of equations, e.g., for x_j, because $x_{j+1} = \bar{x}_j$. We note that all poles, except possibly one, of $[n/n]$ Padé approximants of e^t have nonvanishing imaginary part.

The presence of complex conjugate poles of rational functions $r_{m,n}$ with real coefficients can also be exploited in the determination and evaluation of IPF(τ) representations by requiring that complex conjugate poles and zeros belong to the same partial fraction. We note, however, that complex conjugate poles with a small imaginary part may yield large partial fraction coefficients. Consider the partial fraction representation

$$\frac{1}{(z-t_j)(z-\bar{t}_j)} = \frac{(2\operatorname{Im} t_j)^{-1}}{z-t_j} - \frac{(2\operatorname{Im} t_j)^{-1}}{z-\bar{t}_j}.$$

Thus, if $\operatorname{Im} t_j > 0$ is "tiny" then the partial fraction coefficients are "huge" and may cause loss of accuracy.

Algorithms 3.8–3.10 can be modified so they determine incomplete partial fraction representations that allow exploitation of complex conjugacy. We have omitted a description of this modification in Section 3 in order to keep the presentation of the algorithms as simple as possible. However, our performance comparison includes timings for a code for the evaluation of IPF(∞) representations in which savings in computational work due to complex conjugacy are exploited.

Example 5.8. This example shows timings for the evaluation of IPF(τ) representations. Let the matrix A and vector b be the same as in Example 5.6. We measured the wall-clock time for loop 2 of Algorithm 3.1 since the remaining parts of the computation (defining the matrix A and vector b, and computation of the partial fraction coefficients) have to be carried out only once, and hence their costs are easily amortized when time stepping. The time step δ was chosen to be $4 \cdot 10^{-5}$ for this experiment, although we note that in the context of direct solvers timings are independent of δ. The linear systems of equations were solved by a direct tridiagonal solver. The timings measured are displayed in Fig. 1.

As expected, the time required for IPF(0) representations increases linearly with n, and the fastest integration method is obtained from IPF(∞) representations when complex conjugacy is exploited. However, the tables above show that the accuracy obtained with the IPF(∞) representations is unacceptably low when n is large. On the other hand, using reasonably large values of τ yields good accuracy and wall-clock execution times that are far superior to those for the IPF(0) representations. Comparing Fig. 1 and Table 6, we observe that the sudden upward turns in the curves of Fig. 1 are due to an increase in the number of components μ in the IPF(τ) representation used. The advantage of parallel processing is manifested by the very slow increase with n of the wall-clock times when μ is kept fixed.

From Algorithm 3.1 it is clear that the use of $\mu > 1$ terms to control the size of the partial fraction coefficients entails a sequentialization of part of the computation. When there are enough processors to complete Step 2.1 of the algorithm in parallel, the leading cost of Algorithm 3.1 is μ times the

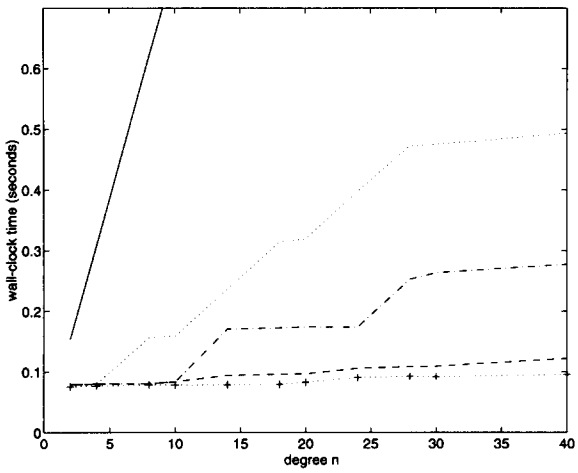

Fig. 1. Wall-clock time required for the evaluation of IPF(τ) representations of $[n/n]$ Padé approximants of $e^{-\delta A}b$, for several values of r and n, and $\delta := 4 \cdot 10^{-5}$. Curves are marked as follows: — ($r = 0$); ··· ($r = 10^4$); — · — · ($\tau = 10^8$); — — ($\tau = \infty$); + ··· ($\tau = \infty$ exploiting conjugacy). See Table 6 for the associated values of μ and errors.

cost of Step 2.1. Hence, in the context of the time stepping schemes of the present section, the costs of performing one time step with an IPF(τ) representation implemented with μ components, and performing μ time steps with an IPF(∞) representation are the same. Remembering that for a fixed accuracy, higher order of approximation allows a larger value of δ, but could also require a larger value of μ, it is important to study how μ grows as δ is increased. For fixed accuracy requirements, we can consider both μ and n to be functions of δ, i.e., $\mu = \mu(\delta)$ and $n = n(\delta)$. In order to minimize the total work required to integrate to the end point, it is desirable that $\mu(\delta)/\delta$ decreases as δ increases. Table 6, for example, tells us that in order to make the base-10 logarithm of the maximum relative error equal to -5.5, we could either use $n = 14$, a complete partial fractions ($\mu = 1$), and $\delta = 1 \cdot 10^{-5}$, or we could use $n = 28$, an IPF(10^8) representation ($\mu = 3$), and $\delta = 4 \cdot 10^{-5}$. It thus seems advantageous to use the latter scheme. This result is corroborated by timings on the Alliant. The former scheme requires 0.09 s wall-clock time to advance one time step, whereas the latter takes 0.25 s wall-clock time to advance a time step four times as large.

Example 5.9. The experiments for this example were conducted in order to investigate the efficiency of the IPF representations for integration of differential equations. In particular, we are interested in the relation between $\mu = \mu(\delta)$, δ and the total runtime (wall-clock time) required for fixed accuracy requirements. We used values of n ranging from 4 to 30, and let the time step vary from $\delta = 1 \cdot 10^{-6}$ up to $\delta = 30 \cdot 10^{-6}$. For each n, we used several values of τ and this generated IPF(τ) representations with different numbers of factors.

In order to achieve eight digits of relative accuracy, measured as in Example 5.7, the diagonal scheme of highest order that we can use with a complete partial fraction representation is based on the [14/14] Padé approximant of e^t, with a maximum time step $\delta_{\text{base}} := 6\delta_{\text{unit}}$ where $\delta_{\text{unit}} = 1 \cdot 10^{-6}$. We refer to this scheme as the "base scheme".

Table 7
Degree n, maximum time step δ in multiples of δ_{unit}, number of components in the IPF representation of $[n/n]$ Padé approximant of $e^{-A\delta}$, run time on an Alliant computer to compute $e^{-A\delta}b$, and efficiency indicators

Degree n	Time step in multiples of δ_{unit}: $\delta/\delta_{\text{unit}}$	# factors $\mu(\delta)$	$(\mu(\delta)/\delta)\delta_{\text{unit}}$	Wall-clock (wc) time (s)	Wall-clock time per unit time step: wc $\delta_{\text{unit}}/\delta$
14	6	1	0.167	0.10	0.0166
18	10	2	0.200	0.18	0.0180
20	12	2	0.167	0.19	0.0158
22	15	2	0.133	0.18	0.0120
24	18	2	0.111	0.18	0.0100
28	25	3	0.120	0.27	0.0108
30	29	3	0.103	0.28	0.0096

For each value of n, we chose the maximum time step δ that yields at least the same accuracy as the base scheme and tabulated its value, as well as the smallest among the corresponding values of $\mu = \mu(\delta)$. Results are presented in Table 7, which also displays $\mu(\delta)/\delta$. The last two columns show wall-clock times for the Alliant computer.

From Table 7 we conclude that, for large values of n, the time step δ increases by a larger factor than $\mu = \mu(\delta)$. Comparing the first with the last rows, we see that δ can increase approximately five times whereas $\mu(\delta)$ increases only by a factor 3. Thus, we expect an integration method based on the [30/30] Padé approximant to be approximately $\frac{5}{3}$ times faster per time step than an integration method based on the [14/14] Padé approximant. The last column of the table shows that the actual run times are in agreement with this prediction.

5.3. Comments on related integration methods

It is interesting to compare the previous results for diagonal Padé approximation with those obtained using rational Chebyshev approximants of the exponential function. The latter methods were pioneered by Varga in [8–10, 53, 55]. Low order integration methods based on rational Chebyshev approximants have been used for a long time, and high-order methods are considered in [18, 20, 21]. Table B.1 of [21] shows partial fraction coefficients for rational Chebyshev approximants $r_{n,n}$ for $n = 10$ and $n = 14$.[4] The table shows that the partial fraction coefficients remain reasonably small for these approximants even for fairly large values of n. For example, the magnitude of the partial fraction coefficient of $r_{n,n}$ of largest magnitude is 21.59 for $n = 10$, and 105.88 for $n = 14$.

In an effort to avoid complex arithmetic, rational approximants of e^t with only real poles are frequently used in integration formulas. Among these rational approximants, those with a single real pole of high multiplicity are popular because they require only the factorization of one real

[4] There is a typographical error in Table B.1 of [21]: It lists the partial fraction coefficients doubled, i.e. $\{2\alpha_j\}_{1 \leq j \leq n/2}$ in accordance to formula B.1.

linear system of equations in each time step, and because they approximate e^t better than rational approximants with multiple real poles; see [24, 34] for a discussion. Therefore, integration methods based on rational approximants with only one pole are attractive to implement on sequential computers. However, they are difficult to implement efficiently on parallel computers.

In order to obtain integration methods that lend themselves well to parallel computation, Serbin [46] recently proposed to first determine rational approximants of e^t with a single pole of high multiplicity, and then perturb the single pole to create several distinct poles. The new rational approximants obtained have partial fraction representations of the form (2), and this makes parallel implementation of the associated integration methods possible, at least if we ignore the sizes of the partial fraction coefficients. However, the closeness of the poles of the rational approximants obtained generally gives rise to partial fraction coefficients of large magnitude, and this can lead to cancellation of significant digits in the computed solution to the differential equation being integrated. This problem was already noted in [46]. The IPF representations of the present paper would appear to be suitable for use in the context of Serbin's integration schemes, since they may be able to restore the accuracy lost due to close distinct poles.

6. Conclusion

We have described algorithms for computing IPF representations of rational functions, and have demonstrated some of their properties. These representations appear well suited for use in a multiprocessor environment, in that they allow parallel computation and yield high accuracy. Applications to integration methods for differential equations appear promising.

Acknowledgements

We would like to thank Zhaojun Bai, Valeria Simoncini and Richard Varga for comments on previous versions of the paper.

References

[1] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov and D. Sorensen, *LAPACK Users' Guide* (SIAM, Philadelphia, 1992).

[2] T. Bagby, On interpolation by rational functions, *Duke Math. J.* **36** (1969) 95–103.

[3] R.E. Bank and D.J. Rose, Marching algorithms for elliptic boundary value problems I: The constant coefficient case, *SIAM J. Numer. Anal.* **14** (1977) 792–829.

[4] G. Birkhoff and R.S. Varga, Discretization errors for well-set Cauchy problems I, *J. Math. Phys.* **44** (1965) 1–23.

[5] C.H. Bischof, B.N. Datta and A. Purkayastha, A parallel algorithm for the multi-input Sylvester-observer equation, Tech. Rep. MCS-P274-119, Math. Sci. Division, Argonne National Lab., Argonne, 1991.

[6] B.L. Buzbee, G.H. Golub and C.W. Nielson, On direct methods for solving Poisson's equation, *SIAM J. Numer. Anal.* **7** (1970) 627–656.

[7] D. Calvetti, E. Gallopoulos and L. Reichel, Accuracy control for parallel evaluation of matrix rational functions, in: R.F. Sincovec, D.E. Keyes, M.R. Leuze, L.R. Petzold and D.A. Reed, Eds., *Proc. 6th SIAM Conf. on Parallel Processing for Scientific Computing* (SIAM, Philadelphia, 1993) 652–655.

[8] A.J. Carpenter, A. Ruttan and R.S. Varga, Extended numerical computations on the $\frac{1}{9}$ conjecture in rational approximation theory, in: P.R. Graves-Morris, E.B. Saff and R.S. Varga, Eds., *Rational Approximation and Interpolation*, Lecture Notes in Math. **1105** (Springer, Berlin, 1984) 383–411.

[9] J.C. Cavendish, W.E. Culham and R.S. Varga, A comparison of Crank–Nicolson and Chebyshev rational methods for numerically solving parabolic equations, *J. Comput. Phys.* **10** (1972) 354–368.

[10] W.J. Cody, G. Meinardus and R.S. Varga, Chebyshev rational approximations to e^{-x} in $[0, +\infty)$ and applications to heat-conduction problems, *J. Approx. Theory* **2** (1969) 50–65.

[11] B.N. Datta and Y. Saad, Arnoldi methods for large Sylvester-like observer matrix equations, and an associated algorithm for partial spectrum assignment, *Linear Algebra Appl.* **154–156** (1991) 225–244.

[12] A. Edrei, Sur les déterminants récurrents et les singularités d'une fonction donnée par son développement de Taylor, *Composito Math.* **7** (1939) 20–88.

[13] B. Fischer and L. Reichel, Newton interpolation in Fejér and Chebyshev points, *Math. Comp.* **53** (1989) 265–278.

[14] R.W. Freund, Solution of shifted linear systems by quasi-minimal residual methods, in: L. Reichel, A. Ruttan and R.S. Varga, Eds., *Numerical Linear Algebra* (De Gruyter, Berlin, 1993) 101–121.

[15] D. Gaier, *Lectures on Complex Approximation* (Birkhäuser, Boston, 1987).

[16] E. Gallopoulos and Y. Saad, Parallel block cyclic reduction algorithm for the fast solution of elliptic equations, Tech. Rep. 659, Center for Supercomputing Research and Development, University of Illinois, Urbana-Champaign, IL, 1987.

[17] E. Gallopoulos and Y. Saad, Parallel block cyclic reduction algorithm for the fast solution of elliptic equations, *Parallel Comput.* **10** (1989) 143–160.

[18] E. Gallopoulos and Y. Saad, On the parallel solution of parabolic equations, in: *Proc. ACM Internat. Conf. on Supercomputing*, Herakleion, Greece (1989) 17–28.

[19] E. Gallopoulos and Y. Saad, Some fast elliptic solvers for parallel architectures and their complexities, *Internat. J. High Speed Computing* **1** (1989) 113–141.

[20] E. Gallopoulos and Y. Saad, Efficient parallel solution of parabolic equations: implicit methods on the Cedar multi-cluster, in: J. Dongarra, P. Messina, D.C. Sorensen and R.G. Voigt, Eds., *Proc. 4th SIAM Conf. on Parallel Processing for Scientific Computing* (SIAM, Philadelphia, 1990) 251–256.

[21] E. Gallopoulos and Y. Saad, Efficient solution of parabolic equations by Krylov approximation methods, *SIAM J. Sci. Statist. Comput.* **13** (1992) 1236–1264.

[22] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II. Stiff and Differential-Algebraic Problems* (Springer, Berlin, 1991).

[23] P. Henrici, An algorithm for the incomplete partial fraction decomposition of a rational function into partial fractions, *Z. Angew. Math. Phys.* **22** (1971) 751–755.

[24] A. Iserles and S.P. Nørsett, *Order Stars* (Chapman & Hall, London, 1991).

[25] O.A. Karakashian and W. Rust, On the parallel implementation of implicit Runge–Kutta methods, *SIAM J. Sci. Statist. Comput.* **9** (1988) 1085–1090.

[26] A.Q.M. Khaliq, E.H. Twizell and D.A. Voss, On parallel algorithms for semidiscretized parabolic partial differential equations based on subdiagonal Padé approximations, *Numer. Methods Partial Differential Equations* **9** (1993) 107–116.

[27] J. Konicek, T. Tilton, A. Veidenbaum, C. Zhu, E. Davidson, R. Downing, M. Haney, M. Sharma, P. Yew, P. Farmwald, D. Kuck, D. Lavery, R. Lindsey, D. Pointer, J. Andrews, T. Beck, T. Murphy, S. Turner and N. Warter, The organization of the Ceder system, in: *Proc. Internat. Conf. on Parallel Processing*. Vol. 1, St. Charles, IL (1991) 49–56.

[28] H.T. Kung, New algorithms and lower bounds for the parallel evaluation of certain rational expressions and recurrences, *J. ACM* **23** (1976) 252–261.

[29] J.D. Lawson and D.A. Swayne, High-order near best uniform approximations to the solution of heat conduction problems, in: *Proc. IFIP Cong.* 80 (North-Holland, Amsterdam, 1980) 741–746.

[30] J.D. Lawson, S.J. Thomas and R.V.M. Zahar, Subspace projection methods for differential systems, Report, Département d'Informatique et de Recherche Operationelle, Université de Montréal, Montréal, 1993.
[31] F. Leja, Sur certaines suites liées aux ensembles plans er leur application à la représentation conforme, *Ann. Polon. Math.* **4** (1957) 8–13.
[32] N. Levenberg and L. Reichel, A generalized ADI iterative method, *Numer. Math.* **66** (1993) 215–233.
[33] S.P. Nørsett, Restricted Padé approximations to the exponential function, *SIAM J. Numer. Anal.* **15** (1978) 1008–1029.
[34] B. Orel, Real pole approximations to the exponential function, *BIT* **31** (1991) 144–159.
[35] P. Pandey, C. Kenney and A.J. Laub, A parallel algorithm for the matrix sign function, *Internat. J. High Speed Comput.* **2** (1990) 181–191.
[36] L. Reichel, The ordering of tridiagonal matrices in the cyclic reduction method for Poisson's equation, *Numer. Math.* **56** (1989) 215–228.
[37] L. Reichel, Newton interpolation at Leja points, *BIT* **30** (1990) 332–346.
[38] L. Reichel, The application of Leja points to Richardson iteration and polynomial preconditioning. *Linear Algebra Appl.* **154–156** (1991) 389–414.
[39] M.F. Reusch, L. Ratzan, N. Pomphrey and W. Park, Diagonal Padé approximations for initial value problems, *SIAM J. Sci. Statist. Comput.* **9** (1988) 829–838.
[40] J.R. Rice, On the conditioning of polynomials and rational forms, *Numer. Math.* **7** (1965) 426–435.
[41] A. Ruhe, Rational Krylov algorithms for nonsymmetric eigenvalue problems, in: G. Golub, A. Greenbaum and M. Luskin, Eds., *Recent Advances in Iterative Methods* (Springer, New York, 1993) 149–164.
[42] E.B. Saff and R.S. Varga, Zeros and poles of Padé approximants to e^z, *Numer. Math.* **25** (1975) 1–14.
[43] E.B. Saff and R.S. Varga, On the zeros and poles of Padé approximants to e^z, II, in: E.B. Saff and R.S. Varga, Eds., *Padé and Rational Approximations: Theory and Applications* (Academic Press, New York, 1977) 195–213.
[44] E.B. Saff and R.S. Varga, On the zeros and poles of Padé approximants to e^z, III, *Numer. Math.* **30** (1978) 241–266.
[45] S.A. Sander, Domain decomposition and rational approximation problems, in: R. Beauwens and P. de Groen, Eds., *Iterative Methods in Linear Algebra* (North-Holland, Amsterdam, 1992) 541–550.
[46] S.M. Serbin, A scheme for parallelizing certain algorithms for the linear inhomogeneous heat equation, *SIAM J. Sci. Statist. Comput.* **13** (1992) 449–458.
[47] P.N. Swarztrauber, A direct method for the discrete solution of separable elliptic equations, *SIAM J. Numer. Anal.* **11** (1974) 1136–1150.
[48] P.N. Swarztrauber and R.A. Sweet, Vector and parallel methods for the direct solution of Poisson's equation, *J. Comput. Appl. Math.* **27** (1989) 241–263.
[49] D.A. Swayne, Matrix operations with rational functions, in: *Proc. 7th Manitoba Conf. on Numerical Mathematics*, Winnipeg, Manitoba (1977) *Utilitas Math.* 581–589.
[50] R.A. Sweet, A parallel and vector variant of the cyclic reduction algorithm. *Internat. J. Supercomputer Appl.* **22** (1987) 10–25.
[51] R.A. Sweet, A parallel and vector cyclic reduction algorithm, *SIAM J. Sci. Statist. Comput.* **9** (1988) 761–765.
[52] M. Tsuji, *Potential Theory in Modern Function Theory* (Maruzen, Tokyo, 1959).
[53] R.S. Varga, On higher order stable implicit methods for solving parabolic partial differential equations, *J. Math. Phys.* **40** (1961) 220–231.
[54] R.S. Varga, *Matrix Iterative Analysis* (Prentice-Hall, Englewood Cliffs, NJ, 1962).
[55] R.S. Varga, Chebyshev semi-discrete approximations for linear parabolic problems, in: P.L. Butzer, J.-P. Kahane and B. Sz.-Nagy, Eds., *Linear Operators and Approximation* (Birkhäuser, Basel, 1972) 452–460.
[56] R.S. Varga, *Scientific Computation on Mathematical Problems and Conjectures* (SIAM, Philadelphia, 1990).
[57] J.L. Walsh, *Interpolation and Approximation by Rational Functions in the Complex Plane* (Amer. Math. Soc., Providence, RI, 4th ed., 1965).
[58] G. Wanner, E. Hairer and S.P. Nørsett, Order stars and stability theorems, *BIT* **18** (1978) 474–489.
[59] D. Westreich, Partial fraction expansion without derivative evaluation, *IEEE Trans. Circuit Systems* **38** (1991) 658–660.

[60] D.M. Young, The search for "high-order" parallelism for iterative sparse linear system solvers, in: G.F. Carey, Ed., *Parallel Supercomputing: Methods, Algorithms and Applications.* (Wiley, Chichester, 1989) 89–106.

[61] D.M. Young and B.R. Vona, On the use of rational iterative methods for solving large sparse linear systems, *Appl. Numer. Math.* **10** (1992) 261–278.

[62] V. Zakian, Application of J_{MN} approximants to numerical initial-value problems in linear differential-algebraic systems, *J. Inst. Math. Appl.* **15** (1975) 267–272.

[63] V. Zakian, Properties of I_{MN} and J_{MN} approximants and applications to numerical inversion of Laplace transforms and initial value problems, *J. Math. Anal. Appl.* **50** (1975) 191–222.

[64] V. Zakian and M. Edwards, Tabulation of constants for full grade I_{MN} approximants, *Math. Comp.* **32** (1978) 519–531.